Class06

Andres Rivero

10/15/2021

Quick Rmarkdown intro

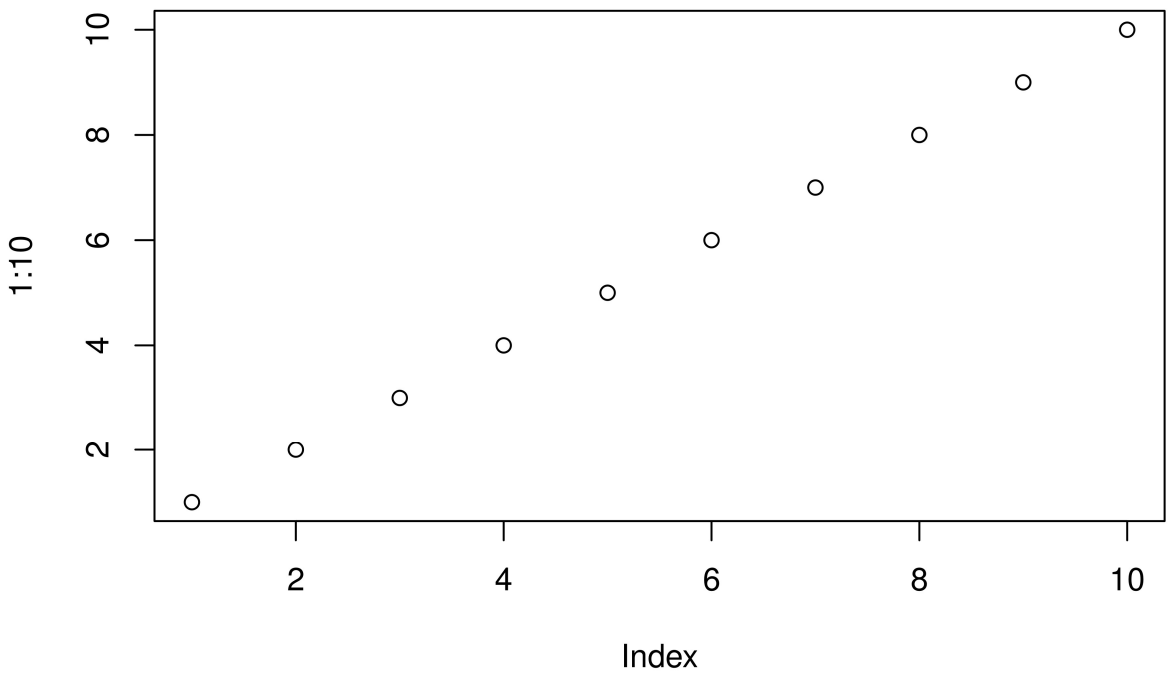
We can write test of course just like any file. **We can style text to be bold** or *italic*.

Do:
- this - and that - and another thing

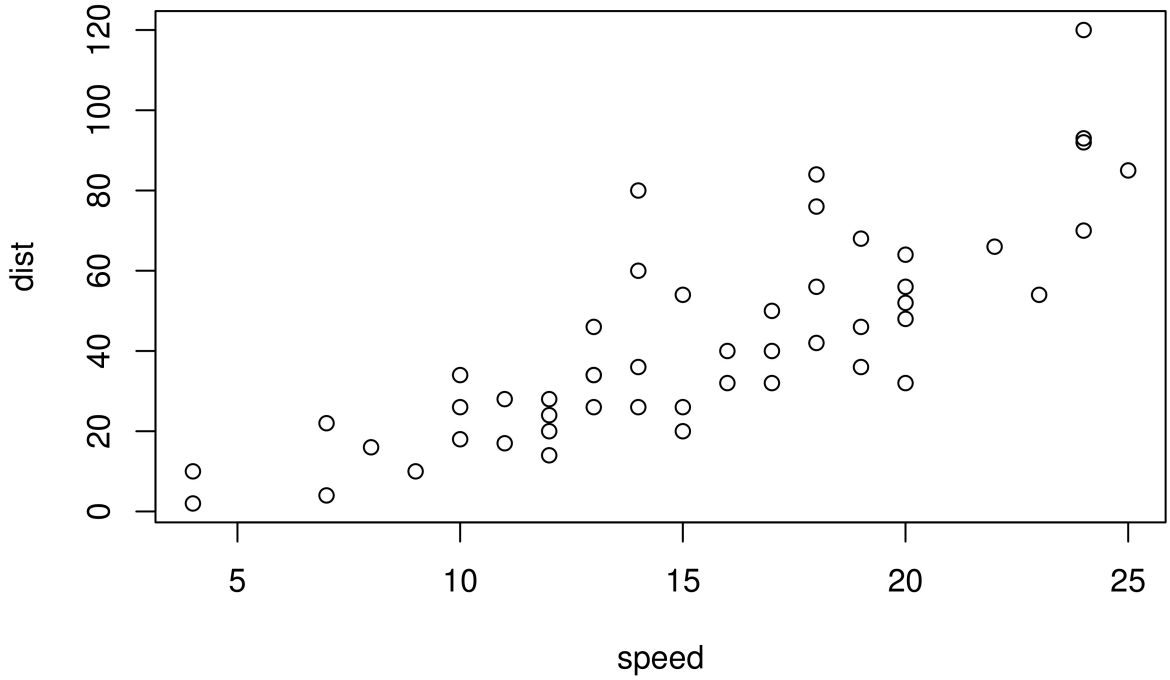
This is more text
and this is a new line

We can include some code:

```
plot(1:10)
```



```
#This is a comment and will not be passed to R
plot(cars)
```




```
# Ctrl+ALt+I
```

Time to write a function

> **Q1**. Write a function grade() to determine an overall grade from a vector of student homework assignment scores dropping the lowest single score. If a student misses a homework (i.e. has an NA value) this can be used as a score to be potentially dropped. Your final function should be adquately explained with code comments and be able to work on an example class gradebook such as this one in CSV format: "https://tinyurl.com/gradeinput" [3pts]

```
# Example input vectors to start with

student1 <- c(100, 100, 100, 100, 100, 100, 100, 90)
student2 <- c(100, NA, 90, 90, 90, 90, 97, 80)
student3 <- c(90, NA, NA, NA, NA, NA, NA, NA)
```

First I want to find the lowest score. I can yse the **min()** to find it and the **which.min() function to find where it is (i.e. its position in the vector).

```
min(student1)
```

```
## [1] 90
```

```
which.min(student1)
```

```
## [1] 8
```

I can use minus to get everything in the vector but the lowest score.

```
student1[-(which.min(student1))]
```

```
## [1] 100 100 100 100 100 100 100
```

Now I can call the **mean()** function to get the average

```
mean(student1[-(which.min(student1))])
```

```
## [1] 100
```

```
mean(student2[-(which.min(student2))])
```

```
## [1] NA
```

NO! But why not?

```
which.min(student2)
```

```
## [1] 8
```

```
mean(student2)
```

```
## [1] NA
```

```
mean(student2, na.rm=TRUE)
```

```
## [1] 91
```

One great idea is to replace the NA values with zero.

```
which(is.na(student2))
```

```
## [1] 2
```

The is.na function returns a logical vector where TRUE elements indicate the presence

```
is.na(student2)
```

```
## [1] FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
```

Lets replace NA with 0

```
student.prime <- student2
student.prime[is.na(student.prime)]=0
student.prime
```

```
## [1] 100   0  90  90  90  90  97  80
```

Putting bits together

```
student.prime <- student2
student.prime[is.na(student.prime)]=0
mean(student.prime[-which.min(student.prime)])
```

```
## [1] 91
```

How about student 3

```
student.prime <- student3
student.prime[is.na(student.prime)]=0
mean(student.prime[-which.min(student.prime)])
```

```
## [1] 12.85714
```

We got it to work, now we simplify it.

```
x <- student3
x[is.na(x)]=0
mean(x[-which.min(x)])
```

```
## [1] 12.85714
```

```
student4 <- c(100, NA, 90, "90", 90, 90, 97, 80)
```

```
as.numeric(student4)
```

```
## [1] 100  NA  90  90  90  90  97  80
```

```
x <- student4
x <- as.numeric(x)
x[is.na(x)]=0
mean(x[-which.min(x)])
```

```
## [1] 91
```

Finally we can write our function:

All functions have at least 3 things.

A name, input arg and a body

Q1

```
grade <- function(x) {
  x <- as.numeric(x)
x[is.na(x)]=0
mean(x[-which.min(x)])
}
```

```
grade(student1)
```

```
## [1] 100
```

For the whole class

To grade the whole class, we read the gradebook for the class.

```
gradebook <- "https://tinyurl.com/gradeinput"
scores <- read.csv(gradebook, row.names = 1)
scores
```

```
##            hw1 hw2 hw3 hw4 hw5
## student-1  100  73 100  88  79
## student-2   85  64  78  89  78
## student-3   83  69  77 100  77
## student-4   88  NA  73 100  76
## student-5   88 100  75  86  79
## student-6   89  78 100  89  77
## student-7   89 100  74  87 100
## student-8   89 100  76  86 100
## student-9   86 100  77  88  77
## student-10  89  72  79  NA  76
## student-11  82  66  78  84 100
## student-12 100  70  75  92 100
## student-13  89 100  76 100  80
## student-14  85 100  77  89  76
## student-15  85  65  76  89  NA
## student-16  92 100  74  89  77
## student-17  88  63 100  86  78
## student-18  91  NA 100  87 100
## student-19  91  68  75  86  79
## student-20  91  68  76  88  76
```

Super useful function= **apply()**. We can use it to grade all our students with our **grade()** function.

```
apply(scores, 1, grade)
```

```
##  student-1  student-2  student-3  student-4  student-5  student-6  student-7
##      91.75      82.50      84.25      84.25      88.25      89.00      94.00
##  student-8  student-9 student-10 student-11 student-12 student-13 student-14
##      93.75      87.75      79.00      86.00      91.75      92.25      87.75
## student-15 student-16 student-17 student-18 student-19 student-20
##      78.75      89.50      88.00      94.50      82.75      82.75
```

Q2 Finding out the student with the highest score:

```
which.max(apply(scores, 1, grade))
```

```
## student-18
##         18
```

```
max(apply(scores, 1, grade))
```

```
## [1] 94.5
```

Q3 Finding out the toughest homework across the class:

```
apply(scores, 2, mean )
```

```
##  hw1  hw2  hw3  hw4  hw5
## 89.0   NA 80.8   NA   NA
```

```
apply(scores, 2, mean, na.rm=TRUE)
```

```
##      hw1      hw2      hw3      hw4      hw5
## 89.00000 80.88889 80.80000 89.63158 83.42105
```

Replace NA values to zero

```
mask <- scores
mask[is.na(mask)]= 0
apply(mask,2, mean)
```

```
##   hw1   hw2   hw3   hw4   hw5
## 89.00 72.80 80.80 85.15 79.25
```

```
which.min(apply(mask,2, mean))
```

```
## hw2
##   2
```

```
min(apply(mask,2, mean))
```

```
## [1] 72.8
```